AD-A277 422

(2)

Final Technical Report

DTIC
S ELECTE D
MAR 23 1994
F

Project Title: Principles of Ultradependability in Chaotic Routing

Principal Investigator: Lawrence Snyder

Award Number: N00014-91-J-1007

Award Dates: 1 October 1990 – 30 September 1993

Award Amount: $364,116.00

94-08989

Introduction

Generally, routers used in state-of-the-art parallel computers are a form of oblivious router known as *dimension order* routers. Examples include the iPSC/2, nCUBE, DELTA, Paragon, Dash, J-Machine, etc. In networks based on these routers all packets between source S and destination D take a single path, oblivious to network congestion and faults. This can lead to poor performance under high loads when congestion can be significant, and to system failure in large or unmaintainable systems where faults are likely. For these two reasons — performance and dependability — adaptive routing techniques have long been advocated.

Adaptive routers can be divided into two broad classes, minimal adaptive, in which packets always take a shortest path to their destinations, and nonminimal adaptive, in which packet routes are not restricted to shortest paths. Though there have been many proposals for both types, it is generally true that nonminimal adaptive routers are superior to minimal adaptive routers [10, 13]. This is consistent with one's intuition as follows: Although a minimal adaptive router can bypass congestion by sending packets along alternate paths, they "discover the congestion too late." That is, when there are no alternate forward paths to choose from, the packet's forward progress is stalled, and is now contributing to the congestion. This affect motivates nonminimal adaptive routing in which a packet can "back up," or *deroute*, and go around congestion when there are no forward paths.

This document has been approved for public release and sale; its distribution is unlimited.

The reason that nonminimal techniques have not been widely used is that they must prevent livelock. (All routers must prevent deadlock, but the techniques are well understood.) Livelock is a condition in which packets continually move around the network, alternately being routed and derouted, but are never delivered. (This cannot happen in oblivious or minimal adaptive algorithms since packets only move forward towards their destinations.) Before the invention of Chaotic routing [12], the only way to prevent livelock was to use priorities or their variant, battle scars. In priority routing, each packet is timestamped as it is injected into the network. Then during routing, the timestamp is checked to assure that the oldest packet in a router is not derouted. This

94 3 21 0

UNIVERSITY OF WASHINGTON
SEATTLE, WASHINGTON 98195

Department of Computer Science & Engineering, FR-35
snyder@cs.washington.edu
(206) 543-9265

March 15, 1994

Defense Technical Information Center
Building 5, Cameron Station
Alexandria, VA 22314

Dear Sir:

Please find enclosed the Final Technical Report for:

Grant No: N00014-91-J-1007

Title: Principles of Ultradependability in Chaotic Routing

PI: Lawrence Snyder

Period: 01 October 1990—30 September 1993

Thank you very much for your support.

Cordially,

[illegible]

Larry Snyder
Professor

LS:jw

guards against a packet circulating forever, since there is a limit on how long it can deroute, i.e. it can never be derouted when it becomes the oldest packet in the network. One difficulty with this scheme is that the circuitry to test the priorities is more complicated than the routing decision, and so greatly slows the router, making it impractical.

Chaotic routing solved the livelock problem simply and elegantly. In Chaotic routing, packets that are derouted are selected at random. This simple expedient eliminates the need for any livelock protection because Chaotic routers are probabilistically livelock free [12]. Although this condition is theoretically weaker than the deterministic livelock freedom that comes with the use of priority-based livelock protection, in practice it is equivalent since the deterministic bound on delivery time is so enormous.

With this simple and elegant solution, Chaotic routers cracked the barrier that prevented other nonminimal adaptive routers from achieving their potential. Specifically, nonminimal adaptive routers could always outperform other routers for heavy loads, where the adaptivity and derouting were more relevant to good performance than raw speed. But at light loads, presumably a common occurrence in parallel computer networks, the raw speed of the router is most important. Livelock protection circuitry slowed other routers to the point of being impractical. Chaos routers, having no livelock protection circuitry, are limited only by the speed of making the adaptive decision. This makes Chaos as fast as any adaptive router, and enables its routing logic to approach the speed of the simple-but-inflexible oblivious routers.

With this perspective, this Ultradependability grant began. There were two objectives for this award:

- Understand the fault tolerance properties of Chaotic routing.
- Determine its practicality relative to oblivious routing.

The research on the second topic was also funded by National Science Foundation grants MIP-9013274 and MIP-9213469. The next two sections treat these topics in detail.

Fault Tolerance of Chaotic Routing

There is a simple intuition used by researchers to support the claim that nonminimal adaptive packet routers are fault tolerant: A failed node appears to packets coming from its neighboring nodes as infinitely congested, and so alternate routes are naturally selected. Though true in principle, it is overly simplistic. First, it ignores the issue of what happened to packets at the router when it failed, and second, it supposes that the failure did not affect the operation of the adjacent nodes. So, the primary research objective was to explain in complete detail the Chaos router's response to faults. (The following discussion draws heavily on the material presented in project publications [6, 8].)

Detection. The first task in fault tolerance is to determine when a fault has occurred. Standard techniques such as parity and checksums are presumed to be included in the physical transmission.

This handles the physical corruption of a channel, and will not be further discussed. At the next level up the task is to recognize when a packet is completely lost or delivered to the wrong destination. In this situation the Chaos router is particularly effective, because as in all adaptive routers packets can arrive at the destination out of order. Thus the Chaos router is forced to tag each packet from source S to destination D with a sequence number to be used to reorder the packets into messages. The condition of a packet becoming "too far" out-of-order can be used as an indication that the packet was lost or possibly it was delivered to the wrong address. The Chaos router signals a "red alert" when an implementation-defined state of "too far" out-of-order is reached.

Out-of-order delivery is not sufficient to identify lost packets or other system faults, so mechanisms such as timeouts are also used. Timeouts are effective because there is a deterministic bound for the time a packet may spend at a single node.

Diagnosis. Once "red alert" has been signaled, no more packets are injected and the system is drained by delivering the packets to their destinations. When system drain is complete, probably determined by a timeout, each communication coprocessor (or processor if there is none) initiates a diagnosis protocol to test neighboring routers and the channels connecting them. The result is information describing the functioning of the network, which is distributed around the network. The protocol checks the information to determine if all components of the system are working. If so, the system automatically resumes execution without any intervention. If not, then the operating system is notified for corrective action.

The reason it is important for the Chaotic router to have the ability to automatically test the correctness of the system is that not all "red alerts" will signal errors. Specifically, the probabilistic livelock freedom property of Chaotic routers implies that there is no worstcase bound on the delivery time of a packet, i.e. packets can be arbitrarily delayed. Though this is extremely rare in practice, it can happen. So, it is not possible for a Chaotic routing system to distinguish between packets that are very slow and those that are lost. Occasionally, therefore, the "red alert" will be signaled when a packet is "too far" out-of-order because it is slow, not because it is lost. In such cases the Chaos network checks itself and if nothing is wrong, resumes operation. Notice that the slow packet has already been delivered as a consequence of the diagnosis protocol.

Recovery. With the faulty components determined, the Chaos network must reconfigure itself to assure correct operation when the execution resumes. There are three issues:

- Disabling destinations.
- Assuring packets are not blocked by failures.
- Disabling channels.

These are considered in turn.

Because Chaotic routers must have a mechanism to tag packets so they can be reordered into messages, it is possible to assure no packets are sent to failed nodes simply by having the ability

☑
☐
☐
des
or
A-1

to trap whenever the tagging mechanism is asked to tag a packet destined for a failed node. This is a simple addition to a communication coprocessor.

If a packet enters a node for which the only productive channel, i.e. a channel that takes it closer to its destination, is faulty, the packet may be immediately derouted, rather than waiting to be randomly chosen to be derouted. Implementing this capability in the standard routing logic of the node is straightforward and only requires that the failed channel be marked during reconfiguration time.

Finally, a channel can be disabled simply by "knocking it out" of the profitable list during the normal operation of the routing logic. This prevents any packet from being directed to a failed channel, and assures correct routing.

The specifics of the protocols are, of course, more complicated than indicated here, but the details can be found in the principal project reports [6, 8]. As the grant concludes, Bill Yost, an MS student in the CSE Department, is implementing the "on chip" components of the Chaos fault tolerance protocol, including the "red alert" signalling, parity, check summing, and timeouts. These should enable a careful test of the fault tolerant capabilities of the router.

Practicality of the Chaos Router

As indicated in the introduction, the Chaos router avoids the "livelock protection" problem characteristic of nonminimal adaptive routers, namely Chaos does not need special circuitry to protect against livelock because Chaotic routing is probabilistically livelock free. This means that the probability that a packet remains in the network for t seconds goes to zero as t increases. Since special livelock protection is not needed, the complexity of Chaos routing can be reduced, making it faster than other nonminimal adaptive routers and as fast as any minimal adaptive router.

However, the principle competition for Chaotic routing is not some other adaptive algorithm, but rather the fast dimension-order routers now in wide use in parallel computers. So the critical question investigated in this research is, "How practical are Chaotic routers compared to dimension-order routers." Here, "practical" has the specific meaning: For a given technology, how close are the node latencies of the two routers?

Node latency is the time required for a packet to move from the input pins of one router to the input pins of the next router. It is the figure of merit because node latency determines the performance of the network under light traffic loads. Under this condition packets are not likely to encounter any congestion and so move unimpeded across the network.[1] Since the two routers are both constrained to perform most operations in the same way and at the same speed, e.g. transmitting the data across the wires or "correcting" the destination, the primary speed determinant will be the difference in the complexity of the two routing decisions.

[1]Performance under moderate and heavy loads is at least as important as performance under light loads, but adaptive routers are well known to be effective under those circumstances. Ironically, the technical challenge has been to make them fast in the noncongested case.

Specifically, when a packet arrives at a node, a dimension order router must (for a mesh, torus or other nonhypercube k-ary d-cube) determine whether the packet "goes straight or turns." For example, for a mesh router that routes first in "X" and then in "Y," the packet goes straight in each dimension, turning only from "X" to "Y" and "turning" for delivery. By contrast, the adaptive decision is more complex. When a packet arrives at a node the adaptive router must select a forward path from among the "productive" channels, i.e. those channels that move the packet closer to its destination. Channels in this set of alternatives are removed from consideration if they are unavailable, usually because they are transmitting a packet, but possibly because of a fault as well. Thus the decision is based on dynamically changing information. Of course, it is precisely the fact that a packet's path can adapt dynamically to congestion that gives the power to adaptive routers. But, it is also a fact that the adaptive decision is necessarily more complex than the oblivious decision.

The challenge and the determinant of the practicality of adaptive routing, then, reduce to the question: How close to the fast oblivious decision can the adaptive decision be made to be? In Bolding's thesis [2] the answer was surprisingly, "within one tick." That is, Bolding[2] demonstrated a 1.2μ CMOS design for a Chaotic router having a node latency of 4 clock cycles at 66 MHz, for a total of 60 ns. This compares with the 3 clock cycle node latency at 66MHz for the oblivious Caltech Elko router, which is based on the same 1.2μ CMOS technology. Thus the difference is only 1 clock cycle. A key fact about these designs is they are both "pad limited," that is, the slowest aspect of the node latency is the switching time of the standard 5V I/O pads. This speed determines the interchip communication time. Thus, in this technology, there is no possibility for improving the cycle time of either router.

The schematic structure of the Chaos router is shown in Figure 1. On the left are five input frames, four from adjacent routers in the cardinal directions, and one from the local processor, called the injection frame. On the right are five output frames, four directed to neighbors and one for delivering packets to the local processor. There are five additional input frames forming the multiqueue. Each frame is capable of holding one 20-flit packet. All frames use the virtual cut-through protocol, i.e. flits of a packet only accumulate in a frame as long as their forward progress is blocked; when they can move forward, they begin vacating the frame immediately without waiting for the frame to fill. A cross-bar interconnects the input and output frames to each other and with the multiqueue frames.

The multiqueue provides local, nonblocking storage for packets that cannot advance. When a packet has fully arrived, i.e. all 20 of its flits are stored in the input frame, it did not have the opportunity to cut-through to any output channel and it no longer does. So, it is directed to a free frame in the multiqueue. This action removes the packet from the channel and allows the adjacent router to transmit another packet. If the multiqueue is full then one of the packets is selected at random for "derouting," i.e. the packet will be sent out the next available channel even if it is not a productive channel, which is the likely case. Packets in the multiqueue take priority in output channel assignment over packets in input frames. Further, multiqueue packets respect time order, i.e. when a multiqueue packet is selected for an output channel, it is the oldest packet that can use that channel.

[2]Other graduate students participated in the design, but the overall responsibility for the success of this effort rests with Bolding.

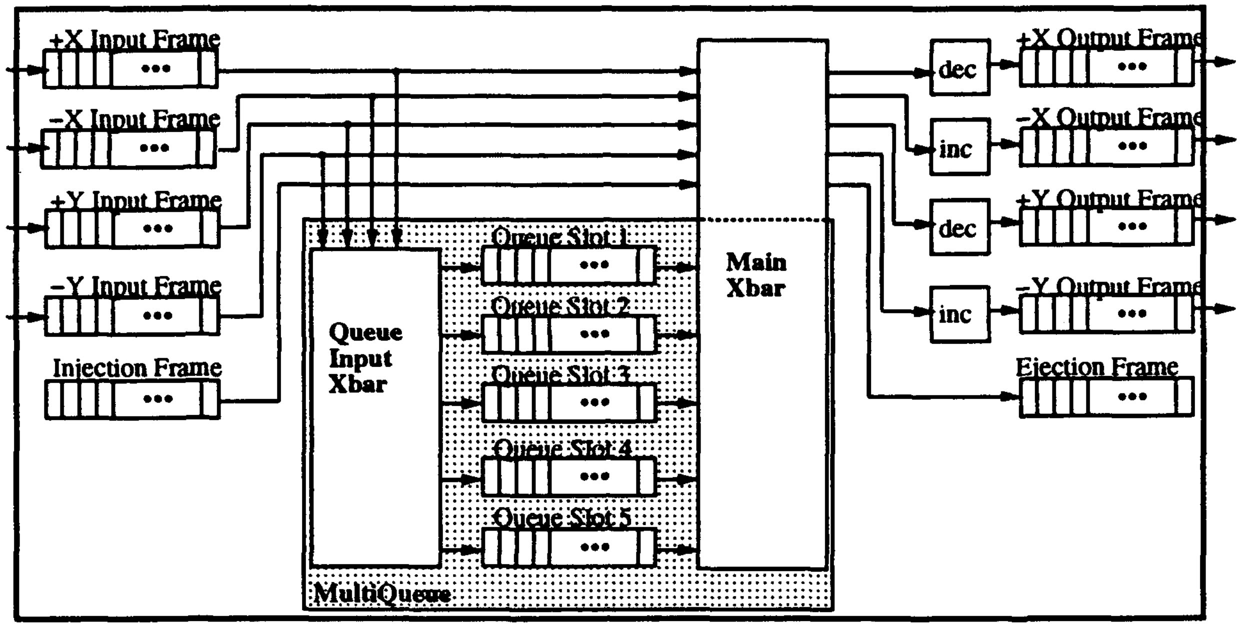

Figure 1: Schematic diagram of a Chaos router. Information moves from left to right. Input frames, output frames and multiqueue frames all use virtual cut-through and are capable of holding one packet.

A packet travels through the router as follows: When a packet header (first flit) arrives at a node, its destination address is decoded, which determines the productive channels. In the next step the channel is chosen and the crossbar is setup. During the third tick the header is transmitted across the cross-bar and the destination address is corrected, i.e. incremented or decremented to reflect the transmission. In the final tick the packet is transmitted across the wires to the next router. Thus, the node latency is "four ticks."

Several points should be noted about the design. First, although multiple packets can arrive at a node simultaneously, it is not wise to process them in parallel for packets having a substantial number of flits. Specifically, the complexity of assigning incoming packets to channels in parallel is sufficiently great, and the probability that several multiflit packets arrive at once in low dimensional networks is sufficiently small [2], that parallel assignment is actually slower [11]. Second, this serial assignment of channels seems to be greatly simplified by an available-channel driven protocol as compared to the obvious arriving-packet driven protocol. Third, the complex operations involving the multiqueue, e.g. selecting the oldest packet that can use a channel or randomly selecting a packet for derouting, are not on the critical path. This means that they need not be solved as rapidly as operations on the critical path.

The design just described was largely developed by Kevin Bolding and presented in his doctoral thesis [2]. The chip was implemented in 1.2μ CMOS using a combination of custom design and Cascade Design Automation standard cells [3]. Approximately 110,000 transistors were used, the die size is 10mm $\times$ 10mm, a 132 pin PGA package was used, and fabrication was brokered by MOSIS.

The resulting chip was tested and founded to be fully functional at 25 MHz. (Higher speed testing is underway.) A single bug has been discovered in the design, which occurs with extreme rarity. Though it causes an extra, fractional packet to be created, an undesirable situation, all packets are nevertheless delivered. A follow-on design, correcting the bug and adding a few enhancements is planned. These enhancements will likely include the fault-tolerance capabilities developed by Bill Yost, and mentioned in the last section.

Other Results

Two other significant studies were conducted under the auspices of this award. They are summarized here.

Virtual Channels

Although Chaotic routing has been shown to work very well on torus-connected networks [7], oblivious routing performs very poorly on the same networks. Although dimension-order routing successfully prevents deadlock on mesh and hypercube networks, extra hardware is necessary to provide deadlock-freedom on torus networks. The virtual-channel scheme of Dally and Seitz provides this deadlock freedom, but at a cost. Because the mechanism that prevents deadlock distinguishes certain nodes as different than others, the uniformity of the torus network is severely upset [1]. This results in significantly reduced performance for obliviously-routed messages in torus networks.

Hypercube vs. Torus

The structure of interconnection networks has always been a cloudy issue in the design of parallel computers. A classic difference has been between hypercube and torus structures. Hypercubes provide small diameters and large bisection bandwidth, but are difficult to build. Tori are easily built, and have wide channels and large local bandwidths, but have large network diameters and poor bisection bandwidth. Most attempts at analyzing networks have either constrained the bisection bandwidth or the number of pins per routing node, arriving at different conclusions depending on the constraint. We proposed a cost model that combines the two factors [5]. Analysis on "equal cost" hypercubes and tori showed that, for random traffic, hypercubes had a slight edge because of their high bisection bandwidth, but, for local traffic, tori performed much better.

Conclusion

When this research was proposed, the Chaos router was a concept with the promise of high performance and fault-tolerance. As a result of this award, and two NSF grants, that promise has been largely demonstrated for both topics.

- A Chaos router with a 4 clock cycle node latency has been designed and implemented, demonstrating that adaptive routing can be realized with nearly the raw speed of oblivious routers, and much higher performance.
- The fault-tolerance concepts for the Chaos router have been worked out, and are being implemented for inclusion in the base chip design.

Since the experimental evidence [4, 9] suggests that no other router has such high throughput and as low a latency across a spectrum of applied loads, the Chaos router is likely to be the router of choice for the next generation of routers.

References

[1] K. Bolding, "Non-Uniformities Introduced by Virtual Channel Deadlock Prevention," Technical Report 92-07-07, University of Washington, July 1992.

[2] K. Bolding, Chaotic Routing – Design and Implementation of an Adaptive Multicomputer Network Router, PhD Thesis, University of Washington, July 1993.*

[3] K. Bolding, S-C. Cheung, S-E. Choi, C. Ebeling, S. Hassoun, T.A. Ngo, and R. Wille, "The Chaos Router Chip: Design and Implementation of an Adaptive Router," *Proceedings of VLSI '93*, pp. 8.2.1–8.2.10, 1993.*

[4] K. Bolding, M. Fulgham, and L. Snyder, "The Case for Chaotic Adaptive Routing," Technical Report 94-02-04, University of Washington, February 1994.*

[5] K. Bolding and S. Konstantinidou, "On The Comparison of Hypercube and Torus Networks," *Proceedings of the 21st International Conference on Parallel Processing,* I:62–66, August 1992.

[6] K. Bolding and L. Snyder, "Overview of Fault Handling for the Chaos Router," *Proceedings of the 1991 IEEE International Workshop on Defect and Fault Tolerance in VLSI Systems,* pp. 124–127, November 1991.*

[7] K. Bolding and L. Snyder, "Mesh and Torus Chaotic Routing," *Advanced Research in VLSI and Parallel Systems: Proceedings of the 1992 Brown/MIT Conference,* pp. 333–347, March 1992.

[8] K. Bolding and L. Snyder, "Network Fault Detection and Recovery in the Chaos Router," *Foundations of Ultradependable Computing,* Vol. II, (ed. G. Koob), Kluwer Academic Publishers, 1994 (to appear).*

[9] M. Fulgham, "Performance of Chaos and Oblivious Routers Under Nonuniform Traffic," Technical Report 93-06-01, University of Washington, June 1993.*

[10] S. Konstantinidou, personal communication, 1988.

[11] S. Konstantinidou, Deterministic and Chaotic Adaptive Routing in Multicomputers, PhD Thesis, University of Washington, May 1991.

[12] S. Konstantinidou and L. Snyder, "The Chaos Router: A Practical Application of Randomization in Network Routing," *Proceedings of the 2nd Symposium on Parallel Algorithms and Architectures*, ACM pp. 21–30, 1990.

[13] T. Nguyen and L. Snyder, "Performance of Minimal Adaptive Routers," *Proceedings of the International Conference on Parallel Processing*, 1994 (to appear).*

* Supported by this award.